Microsoft® Windows® Me

Millennium Edition

Operating System

Understanding Universal Plug and Play

White Paper

Abstract

This paper provides an overview of Universal Plug and Play (UPnP) and how it works. Included are scenarios where UPnP improves the overall networking experience by automatically connecting and controlling various types of devices. Additional detail is given on the components of, and protocols and procedures used in UPnP, focusing on how existing standard protocols are used to build UPnP devices. We will then point to where more information can be found to further your knowledge of UPnP.

Contents

Introduction

What is UPnP?

With the addition of Device Plug and Play (PnP) capabilities to the operating system it became a great deal easier to setup, configure, and add peripherals to a PC. Universal Plug and Play (UPnP) extends this simplicity to include the entire network, enabling discovery and control of devices, including networked devices and services, such as network-attached printers, Internet gateways, and consumer electronics equipment.

UPnP is more than just a simple extension of the Plug and Play peripheral model. It is designed to support zero-configuration, "invisible" networking, and automatic discovery for a breadth of device categories from a wide range of vendors.

With UPnP, a device can dynamically join a network, obtain an IP address, convey its capabilities, and learn about the presence and capabilities of other devices—all automatically; truly enabling zero configuration networks. Devices can subsequently communicate with each other directly; thereby further enabling peer to peer networking.

The varieties of device types that can benefit from a UPnP enabled network are large and include intelligent appliances, wireless devices, and PCs of all form factors.

The scope of UPnP is large enough to encompass many existing, as well as new and exciting scenarios including home automation, printing and imaging, audio/video entertainment, kitchen appliances, automobile networks, and proximity networks in public venues.

UPnP uses standard TCP/IP and Internet protocols, enabling it to seamlessly fit into existing networks. Using these standardized protocols allows UPnP to benefit from a wealth of experience and knowledge, and makes interoperability an inherent feature.

Because UPnP is a distributed, open network architecture, defined by the protocols used, it is independent of any particular operating system, programming language, or physical medium (just like the Internet). UPnP does not specify the APIs applications will use, allowing operating system vendors to create the APIs that will meet their customer needs.

Who Defines UPnP?

The Universal Plug and Play Forum defines UPnP Device and Service Descriptions (originally called Device Control Protocols or DCPs) according to a common device architecture contributed by Microsoft. The Universal Plug and Play Forum is a group of companies and individuals across the industry that intend to play a leading role in the authoring of specifications for UPnP devices and services. Formed on October 18, 1999, it is an association of more than 200 vendors who are industry leaders in consumer electronics, computing,

home automation and security, home appliances, computer networking, and mobile devices.

The goals of the Forum are to enable the emergence of easily connected devices and to simplify the implementation of networks in the home and corporate environments. The Forum will achieve this by defining and publishing UPnP device and service descriptions built on open, Internet-based communication standards.

The Forum's web site, http://upnp.org/, is the central repository for schema that has been developed and standardized by the Forum. In addition, the site includes the device architecture document, templates for device and service descriptions, and guidelines for device and service description design. UPnP.org also distributes information about the Forum's activities and progress.
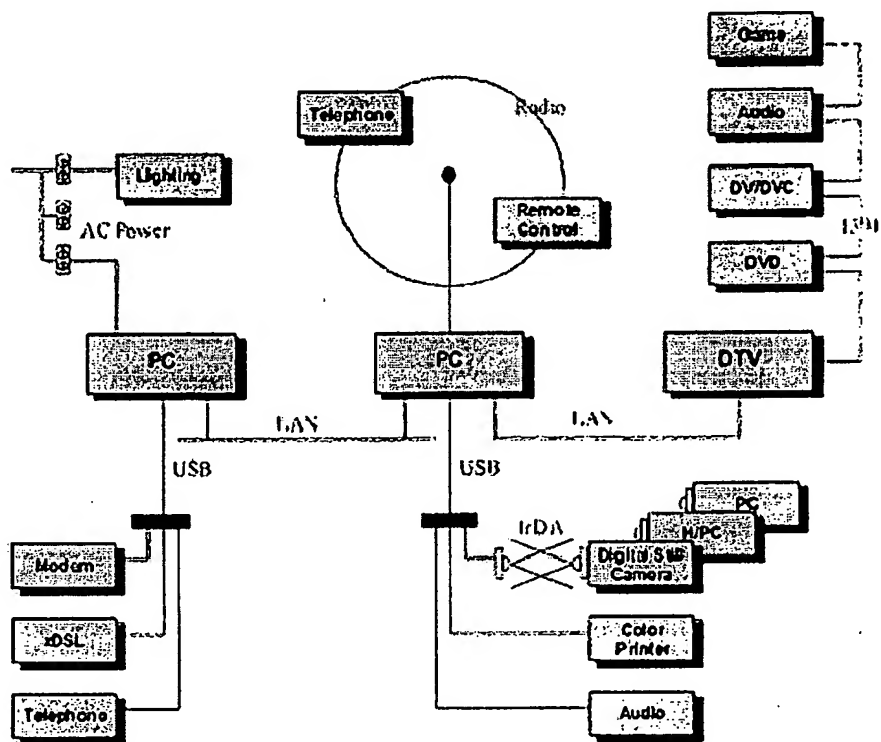
Figure 1: A Mixed Media Multiple Vendor Network

UPNP Enabled Scenarios

UPnP Opens Up a World of Possibilities

There are many ways automatic discovery and control of devices can make life easier and better. The advent of UPnP is opening up new possibilities and uses every day. I dare say that the explosion of possibilities has not been matched since the advent of the computer network. The scenarios seem to be nearly endless. I have included just a few of many ideas in this section.

Some UPnP Scenarios:

The New Printer

If I were to come home today with a new printer to use on my home network, I'd have to take several steps to make sure that the printer could be used by all of the PCs on the network, let alone available to any other device that may have a need to print.

For example, I would need to connect the printer to an existing PC, load the device driver, share the printer on the network, then go to each of the other PCs on the network and connect to that shared printer.

Wouldn't it be nice to simply plug the printer into any available network port, be it phone line, power line, or Ethernet, and have the printer immediately available to all devices and users of the network?

Out of Storage Space Again

I am, ever so often and with increasing frequency, running out of storage space on my PC. I suppose this has something to do with the large number of digital pictures and movies I take, or possibly because I've made a valiant effort to catalog my entire CD collection as Windows Media™ files, but I don't see the situation getting any better soon.

Although the prices for secondary storage have decreased dramatically, the pain of adding a hard drive to one of my systems has only improved marginally through device Plug and Play. Worse yet is trying to figure out how to take those home movies, pictures, and sounds across the country when I go to visit my family on holidays.

Suppose I had a high speed, high capacity, mobile data store that was UpnP-enabled. Perhaps this device connected directly to an Ethernet or phone line network. Or—to enable streaming of my home movies—the device could be connected to the high speed IEEE 1394 bus with the rest of my entertainment system. Wherever I plug in this device, all other devices on the network (including those that produce or display media) immediately know of its availability.

Won't my family be pleased when I visit them with seemingly endless home movies!

New, portable devices aren't limited to storage or printers, but could include many other devices including cameras, MP3 players, scanners, MIDI devices, remote controls, TVs and video devices; the list is seemingly endless.

"It's always freezing when I wake up!"

You're in bed, about to go to sleep and tomorrow is Saturday. Your alarm clock usually wakes you up at 7 A.M., but tomorrow you want to sleep in. You need to be awakened by an alarm, but 9 A.M. seems like a much better time. So you set the alarm to 9 A.M. instead of 7 A.M.

Since you have a UPnP alarm clock, life is good. You have a script running on your Windows®-based PC that is waiting for an alarm notification from the alarm clock. As soon as this happens, it instructs the timer on the HVAC device to set the wake time to the same as the alarm clock time.

Now, your heat will turn on early enough so that you won't freeze when you get out of bed! With an intelligent HVAC system, many other features could be added beyond what the programmable thermostats of today provide. This could include enabling the system when sensors detect people, and remote control of the system over the Internet.

If your alarm clock had access to your schedule information, it could warn you if you choose to wake up after a meeting was scheduled. In other words, if you have a meeting at 9 A.M., and try to set your alarm for 9 A.M., it could warn you that you really ought to get up at 8 A.M. to be able to make it to work by 9.

The Master Switch:

You come home from a hard day's work and walk in your front door. You flip a wall switch, which, to most of us, just turns on the light in the foyer. With UPnP, this switch is much more than that. This switch is simply a UPnP service whose state is defined by a variable called "Position." When you flip the switch, "Position" changes to "On" and a script running on your Windows-based PC goes into action.

The script gets the notification that the position of the master switch has changed to "on". It then does the following:

- The heat turns on to your preset temperature.

- The answering machine starts playing new messages.

- Your stereo system turns on and is set to your favorite classical station, and the volume is set for ambient level.

- The window blinds raise, but only if it's after sunset (sunrise/sunset data is easily obtainable from the Internet).

- Optionally, the TV is turned on and set to the news station, with the sound off and the closed captioning turned on.

- Oh yeah, the light in the foyer also turns on.

Similarly, when you flip the switch to the "off" position, the reverse occurs:
- The heat turns off (or is set at a lower temperature).
- The stereo is turned off.
- The TV is turned off.
- The window blinds lower for privacy.
- All of the lights in the house are turned off.

Synchronize Your Clocks:

It happens all the time. The power goes out for a couple of hours, and every clock in your home is blinking 12:00. Going through the whole house to set them is annoying and time consuming.

Enter UPnP. A script running on your Windows-based PC periodically synchronizes all the clocks in the house to the atomic clock in Colorado. The script looks for all clock services on the network, regardless of the device. It then simply iterates through each service, and sets the time to the value it obtained from the atomic clock over the Internet.

This script can be running as a task that recurs every night at midnight. It can be manually run after the power comes back on to reset each clock.

Vacation Mode

If you were going to be away from your home for an extended period of time, a simple switch to put your home into "vacation" mode would be very useful. A "vacation mode" application could be run on your PC, which would do the following:
- Periodically turn on and off the lights, and even the TV or stereo, in a seemingly random pattern (security).
- Turn the water heater down or off (energy savings).
- Turn the heat down or off (energy savings).
- Lower the blinds (security).
- Turn off all alarm clocks in the house (energy savings).
- Set the VCR to record all of your favorite shows for the time that you are away (convenience).

Until the post-office can be UPnP-enabled, you'll have to stop the mail yourself.

Home theater mode:

When it's time to watch your favorite movie, a single script can automate the experience:

- The TV is turned on to the proper channel (i.e. channel 3).

- The DVD player is turned on.

- The stereo is turned on to the Video mode, and the volume is turned up to theater levels.

- The lights in the house turn off, except for the living room where they dim slightly.

- The DVD is played.

Take Me Out to the Ball Game

Suppose you take your wireless Pocket PC or laptop to the ball game with you. As you travel to or from the game, the device is used in the car to pick up traffic information or roadside service information. This information can then be used to avoid congestion as much as possible, to pick up food and gas, or—if you decided to go at the last minute—to order your tickets and have them waiting at the Will Call booth for you when you arrive.

When you get to the game, your device detects the stadium network and you can automatically sync up to the announcer commentary, order food and souvenirs and track player statistics.

Your buddy list could be searched for any acquaintances that might also have left work early to see the game, and you could chat with them about the particulars of this call or that, even though they ended up in the "cheap seats."

The types of capabilities this scenario describes could easily apply to other events and recreational activities.

Family Togetherness

The entire family has their own personal calendars that they manage with their PDAs, cell phone, office or home PCs that are all UPnP-enabled. The integrated family calendar and the personal calendars are on the home network server.

Each calendar is a UPnP service that can be seen by any member of the family as well as other enabled services. Because the PDAs connect to the wireless home network when they are at home and synchronize the calendar, every member of the family leaves for the day with everyone else's calendar information.

The PDAs can re-synchronize with the home server via cell or direct Internet connections. While at work, mom realizes she needs to stay late for a meeting and will not have time to shop for and cook dinner. She uses the PDA and cell phone to send the grocery list to her favorite store for filling. She then sends all her family members an e-mail update to the family calendar that informs them she will be late. She tells her teenage son to pickup the groceries and start dinner.

Dad checks the family schedule and realizes his son may need help getting all the groceries home and making dinner in time for them to attend a basketball game, so he goes to the grocery store to pickup his son and heads home to make dinner.

The Well Connected Home

It is easy to consider a home in which UPnP makes life easier or richer for many of the day-to-day tasks. Kitchen appliances could be connected to a power line network to form a home network. Information for each of the appliances, such as model, make, and serial number, could be accessed from many locations and retrieved for service calls. A URL for the manufacturer and manual could be accessed for anyone who wants to repair appliances on their own.

Some appliances could be remotely controlled to be turned on or off, have their temperature controlled, or even signal the need for maintenance when they detect an abnormality. Many things would be made possible through remote control, including remotely closing the garage door, remotely checking on a baby in day-care or at home, and remotely programming a VCR to record a TV program you would otherwise miss.

You could have a kitchen tablet on which you could make online product selections as you peruse your cupboards to see what's missing. An attached bar code scanner might make making lists easier. This device would have automatic access to the home network and Internet using UPnP.

The entire home security system could be connected to the UPnP network enabling you to be notified of a breach wherever you may be through the Internet. Sensors in each room to detect your presence could allow music, the baby monitor or video streams to follow you as you roam through the house performing those mundane household duties.

Meeting Time

You're at work. You walk into a conference room where you are scheduled to present your latest proposal. Your wireless-enabled laptop immediately recognizes the devices available in the room and connects to the display device so that your PowerPoint presentation can be displayed. Perhaps you can keep the other wireless attendees from reading email while you talk by displaying the presentation on their laptops also.

Components of a UPnP Network

Devices, Services and Control Points

The basic building blocks of a UPnP network are devices, services and control points. These are described more fully in this section.
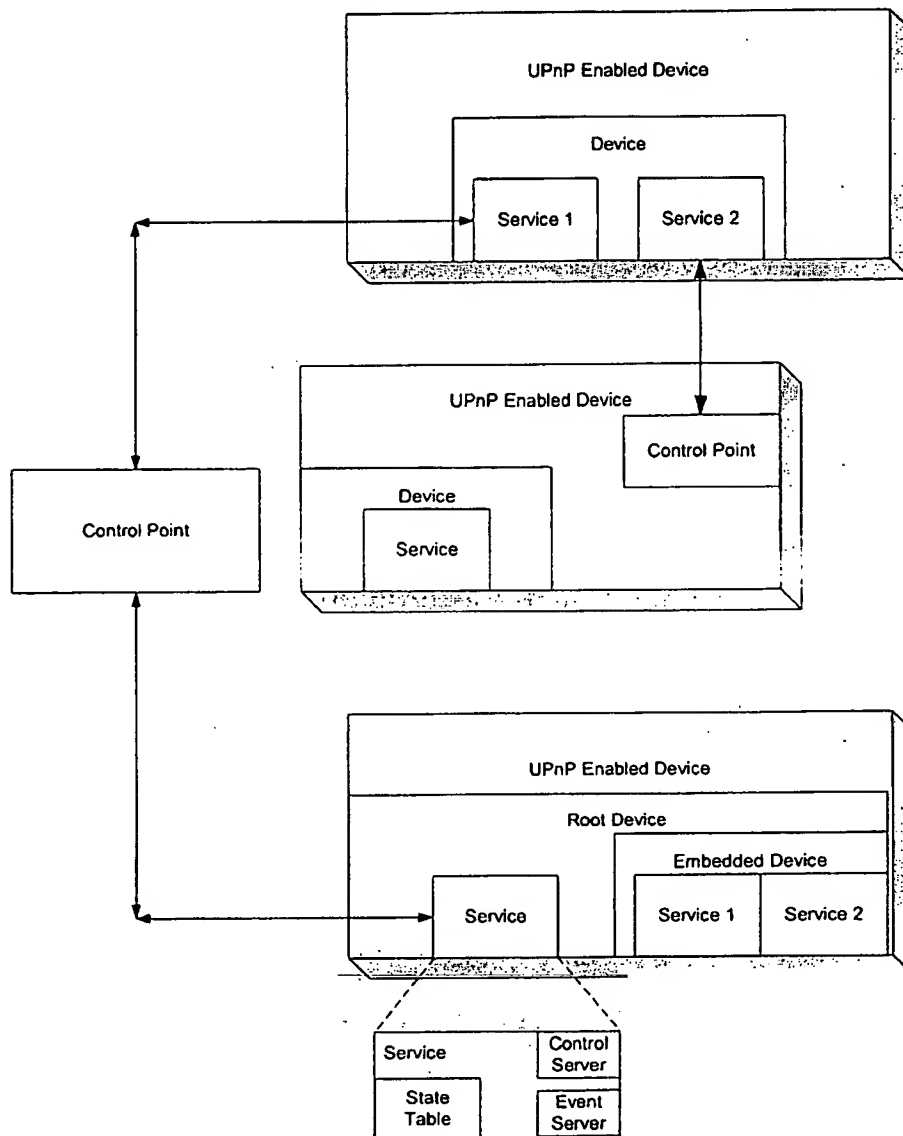

Figure 4: UPnP Control Points, Devices, and Services

Devices

A UPnP device is a container of services and nested devices. For instance, a VCR device may consist of a tape transport service, a tuner service, and a

clock service. A TV/VCR combo device would consist not just of services, but a nested device as well.

Different categories of UPnP devices will be associated with different sets of services and embedded devices. For instance, services within a VCR will be different than those within a printer. Consequently, different working groups will standardize on the set of services that a particular device type will provide. All of this information is captured in an XML device description document that the device must host. In addition to the set of services, the device description also lists the properties (such as device name and icons) associated with the device.

Services

The smallest unit of control in a UPnP network is a service. A service exposes actions and models its state with state variables. For instance, a clock service could be modeled as having a state variable, current_time, which defines the state of the clock, and two actions, set_time and get_time, which allow you to control the service. Similar to the device description, this information is part of an XML service description standardized by the UPnP forum. A pointer (URL) to these service descriptions is contained within the device description document. Devices may contain multiple services.

A service in a UPnP device consists of a state table, a control server and an event server. The state table models the state of the service through state variables and updates them when the state changes. The control server receives action requests (such as set_time), executes them, updates the state table and returns responses. The event server publishes events to interested subscribers anytime the state of the service changes. For instance, the fire alarm service would send an event to interested subscribers when its state changes to "ringing."

Control Points

A control point in a UPnP network is a controller capable of discovering and controlling other devices. After discovery, a control point could:

- Retrieve the device description and get a list of associated services.
- Retrieve service descriptions for interesting services.
- Invoke actions to control the service.
- Subscribe to the service's event source. Anytime the state of the service changes, the event server will send an event to the control point.

It is expected that devices will incorporate control point functionality (and vice-versa) to enable true peer-to-peer networking.

UPnP Protocol Overview

Networking Media for UPnP

UPnP leverages the standard IP protocol suite to remain network media agnostic. Devices on a UPnP network can be connected using any communications media including Radio Frequency (RF, wireless), phone line, power line, IrDA, Ethernet, and IEEE 1394. In other words, any medium that can be used to network devices together can enable UPnP. The only concern might be that the media being used supports the bandwidth required for the intended use.

UPnP uses open, standard protocols such as TCP/IP, HTTP and XML. However, other technologies could be used to network devices together for many reasons, including cost, technology requirements, or legacy support. These include networking technologies like HAVi, CeBus, LonWorks, EIB, or X10. These too can participate in the UPnP network through a UPnP bridge or proxy. A UPnP network containing bridged devices might look like the figure below.
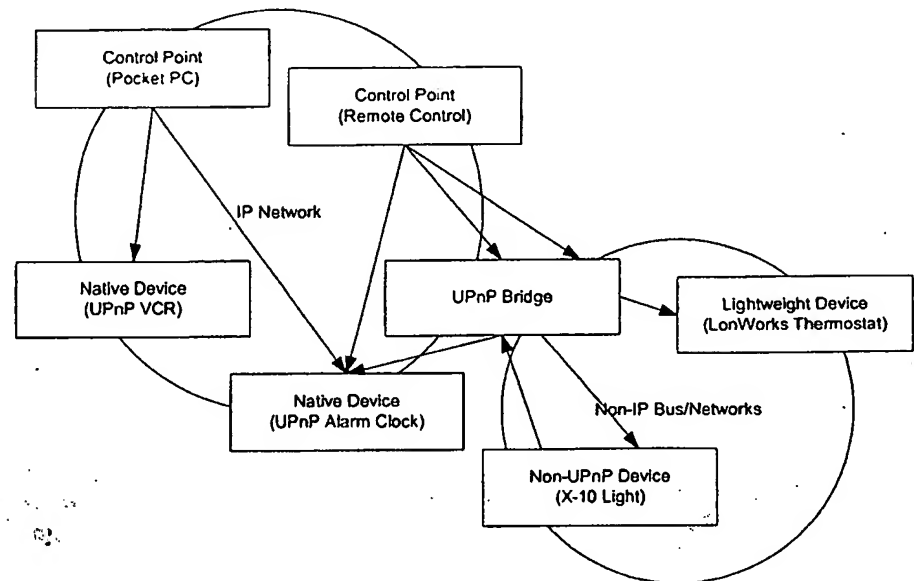
Figure 2 A Bridged UPnP Network

Protocols Used by UPnP

UPnP leverages many existing, standard protocols. Using these standardized protocols aids in ensuring interoperability between vendor implementations.

The protocols used to implement UPnP are found in use on the Internet and on local area networks everywhere. This prevalence ensures that there is a large pool of people knowledgeable in implementing and deploying solutions based on these protocols. Since the same protocols are already in use, little would need to be done to make UPnP devices work in an existing networked environment.

Some of the protocols used to implement UPnP are summarized in the rest of this section.
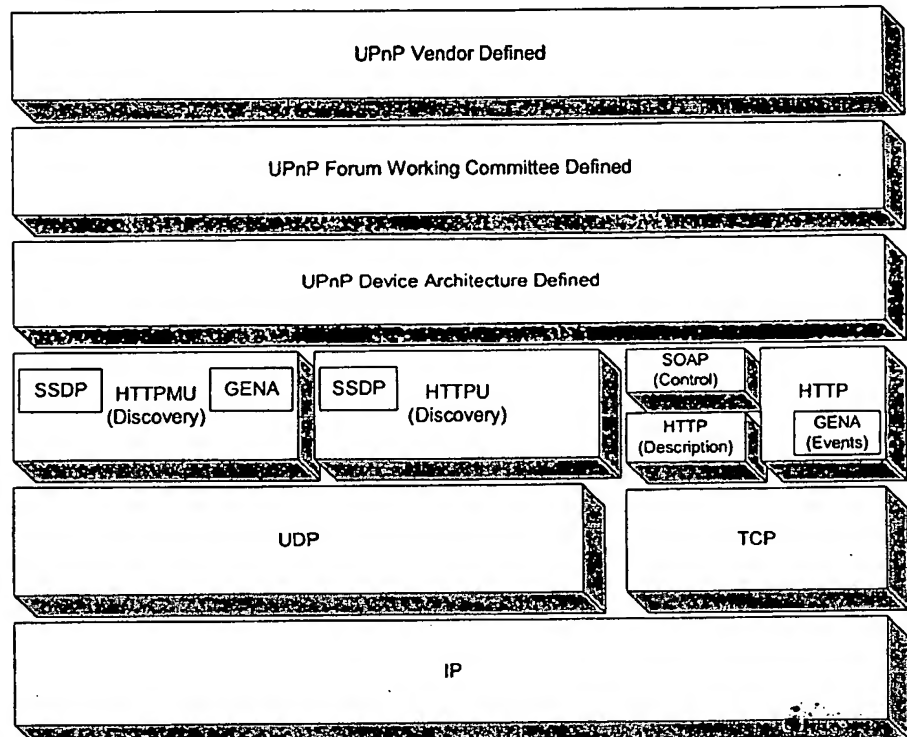
Figure 3: The UPnP Protocol Stack

UPnP Specific Protocols

UPnP vendors, UPnP Forum Working Committees and the UPnP Device Architecture document define the highest layer protocols used to implement UPnP. Based on the device architecture, the working committees define information global to specific device types such as VCRs, HVAC systems, dishwashers, and other appliances. Subsequently, UPnP Device Vendors define the data specific to their devices such as the model name, URL, etc.

TCP/IP

The TCP/IP networking protocol stack serves as the base on which the rest of the UPnP protocols are built. By using the standard, prevalent TCP/IP protocol suite, UPnP leverages the protocol's ability to span different physical media and ensures multiple vendor interoperability.

UPnP devices can use many of the protocols in the TCP/IP stack including TCP, UDP, IGMP, ARP and IP as well as TCP/IP services such as DHCP and DNS. How these protocols and services are used to provide what is required for UPnP to work will become clear as we discuss the other protocols in this section and discuss how UPnP works in subsequent sections.

Since TCP/IP is one of the most ubiquitous networking protocols, locating or creating an implementation for a UPnP device that is tuned for footprint and/or performance is relatively easy.

A basic understanding of the TCP/IP protocol suite and services is assumed in this document. More information on TCP/IP can be found in the references listed at the end of this document.

HTTP, HTTPU, HTTPMU

TCP/IP provides the base protocol stack to provide network connectivity between UPnP devices. HTTP, which is hugely responsible for the success of the Internet, is also a core part of UPnP. All aspects of UPnP build on top of HTTP or its variants.

HTTPU (and HTTPMU) are variants of HTTP defined to deliver messages on top of UDP/IP instead of TCP/IP. These protocols are used by SSDP, described next. The basic message formats used by these protocols adheres with that of HTTP and is required both for multicast communication and when message delivery does not require the overhead associated with reliability.

Some of the explanations of higher-level protocols and the workings of UPnP assume a basic knowledge of the HTTP protocol. More information on HTTP can be found through the references listed at the end of this document.

SSDP

Simple Service Discovery Protocol (SSDP), as the name implies, defines how network services can be discovered on the network. SSDP is built on HTTPU and HTTPMU and defines methods both for a control point to locate resources of interest on the network, and for devices to announce their availability on the network. By defining the use of both search requests and presence announcements, SSDP eliminates the overhead that would be necessary if only one of these mechanisms is used. As a result, every control point on the network has complete information on network state while keeping network traffic low.

Both control points and devices use SSDP. A UPnP control point, upon booting up, can send an SSDP search request (over HTTPMU), to discover devices and services that are available on the network. The control point can refine the search to find only devices of a particular type(such as a VCR), particular services (such as devices with clock services) or even a particular device.

UPnP devices listen to the multicast port. Upon receiving a search request, the device examines the search criteria to determine if they match. If a match is found, a unicast SSDP (over HTTPU) response is sent to the control point.

Similarly, a device, upon being plugged into the network, will send out multiple SSDP presence announcements advertising the services it supports.

Both presence announcements and unicast device response messages contain a pointer to the location of the device description document, which has information on the set of properties and services supported by the device.

In addition to the discovery capabilities provided, SSDP also provides a way for a device and associated service(s) to gracefully leave the network (bye-bye notification) and includes cache timeouts to purge stale information for self healing.

GENA

Generic Event Notification Architecture (GENA) was defined to provide the ability to send and receive notifications using HTTP over TCP/IP and multicast UDP. GENA also defines the concepts of subscribers and publishers of notifications to enable events.

GENA formats are used in UPnP to create the presence announcements to be sent using Simple Service Discovery Protocol (SSDP) and to provide the ability to signal changes in service state for UPnP eventing. A control point interested in receiving event notifications will subscribe to an event source by sending a request that includes the service of interest, a location to send the events to and a subscription time for the event notification.

The subscription must be renewed periodically to continue to receive notifications, and can also be canceled using GENA.

SOAP

Simple Object Access Protocol (SOAP) defines the use of Extensible Markup Language (XML) and HTTP to execute remote procedure calls. It is becoming the standard for RPC based communication over the Internet. By making use of the Internet's existing infrastructure, it can work effectively with firewalls and proxies. SOAP can also make use of Secure Sockets Layer (SSL) for security and use HTTP's connection management facilities, thereby making distributed communication over the Internet as easy as accessing web pages.

Much like a remote procedure call, UPnP uses SOAP to deliver control messages to devices and return results or errors back to control points.

Each UPnP control request is a SOAP message that contains the action to invoke along with a set of parameters. The response is a soap message as well and contains the status, return value and any return parameters.

XML

Extensible Markup Language (XML), to use the W3C definition, is the universal format for structured data on the Web. Put another way, XML is a way to place nearly any kind of structured data into a text file.

XML looks a lot like HTML in that it uses tags and attributes. Actually, it is quite different in that these tags and attributes are not globally defined as to their meaning, but are interpreted within the context of their use. These features of XML make it a good fit for developing schemas for various document types. The use of XML as a schema language is defined by the W3C.

XML is a core part of UPnP used in device and service descriptions, control messages and eventing.

How UPNP Works

The Responsibilities of UPnP

UPnP provides support for communication between control points and devices. The network media, the TCP/IP protocol suite and HTTP provide basic network connectivity and addressing needed. On top of these open, standard, Internet based protocols, UPnP defines a set of HTTP servers to handle discovery, description, control, events, and presentation.

This section describes how the protocols defined earlier in this paper are used to provide for these needs.

The UPnP Protocol Stack

We've described the protocols used to implement UPnP, but to understand these protocols better, let us look at them in a diagram.
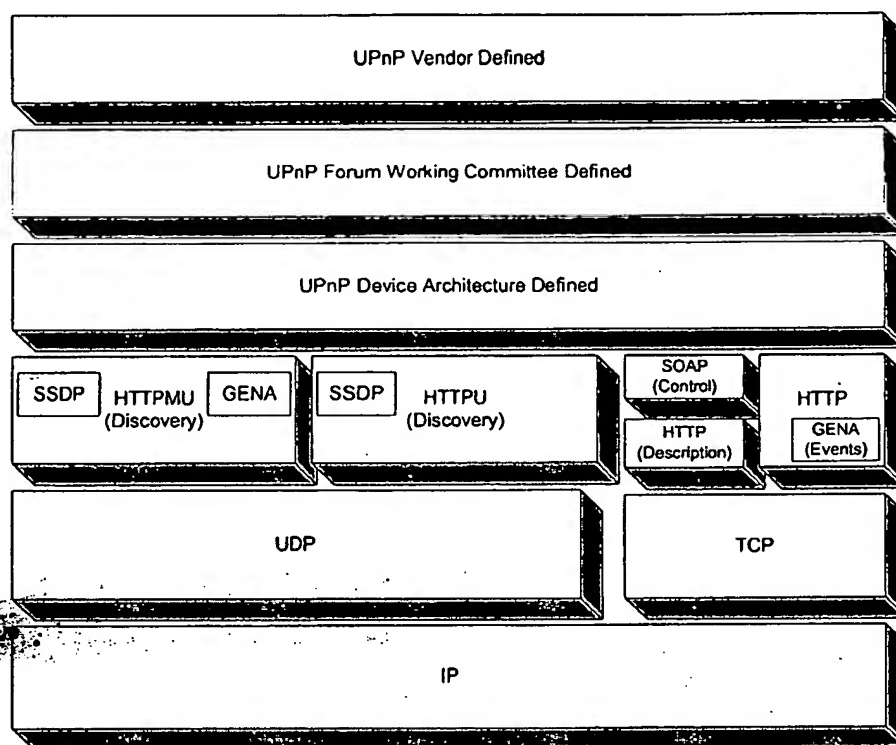

Figure5: The UPnP Protocol Stack

The UPnP Device Architecture defines a schema or template for creating device and service descriptions for any device or service type.

Individual working committees subsequently standardize on various device and service types and create a template for each individual device or service type.

Finally, a vendor fills in this template with information specific to the device or service, such as the device name, model number, manufacturer name and URL to the service description.

This data is then encapsulated in the UPnP-specific protocols defined in the UPnP Device Architecture document (such as the XML device description template).

The required UPnP specific information is inserted into all messages before they are formatted using SSDP, GENA, and SOAP and delivered via HTTP, HTTPU, or HTTPMU.

Steps Involved in UPnP Networking

Addressing

The foundation for UPnP networking is the TCP/IP protocol suite and the key to this suite is addressing. Each device must have a Dynamic Host Configuration Protocol (DHCP) client and search for a DHCP server when the device is first connected to the network. If a DHCP server is available, the device must use the IP address assigned to it. If no DHCP server is available, the device must use Auto IP to get an address.

In brief, Auto IP defines how a device intelligently chooses an IP address from a set of reserved private addresses, and is able to move easily between managed and unmanaged networks.

A device may implement higher layer protocols outside of UPnP that use friendly names for devices. In these cases, it becomes necessary to resolve friendly host (device) names to IP address. Domain Name Services (DNS) are usually used for this. A device that requires or uses this functionality may include a DNS client and may support dynamic DNS registration for its own name to address mapping.

Discovery

Once devices are attached to the network and addressed appropriately, discovery can take place. Discovery is handled by the SSDP as discussed earlier. When a device is added to the network, SSDP allows that device to advertise its services to control points on the network. When a control point is added to the network, SSDP allows that control point to search for devices of interest on the network.

The fundamental exchange in both cases is a discovery message containing a few, essential specifics about the device or one of its services, for example its type, identifier, and a pointer to its XML device description document.

Description

The next step in UPnP networking is description. After a control point has discovered a device, the control point still knows very little about the device. For the control point to learn more about the device and its capabilities, or to interact with the device, the control point must retrieve the device's description from the URL provided by the device in the discovery message.

Devices may contain other, logical devices and services. The UPnP description for a device is expressed in XML and includes vendor-specific, manufacturer information including the model name and number, serial number, manufacturer name, URLs to vendor-specific Web sites, and so forth. The description also includes a list of any embedded devices or services, as well as URLs for control, eventing, and presentation.

Control

After a control point has retrieved a description of the device, the control point has the essentials for device control. To learn more about the service, a control point must retrieve a detailed UPnP description for each service. The description for a service is also expressed in XML and includes a list of the commands, or actions, the service responds to, and parameters or arguments, for each action. The description for a service also includes a list of variables; these variables model the state of the service at run time, and are described in terms of their data type, range, and event characteristics.

To control a device, a control point sends an action request to a device's service. To do this, a control point sends a suitable control message to the control URL for the service (provided in the device description). Control messages are also expressed in XML using SOAP.

In response to the control message, the service returns action specific values or fault codes.

Eventing

A UPnP description for a service includes a list of actions the service responds to and a list of variables that model the state of the service at run time. The service publishes updates when these variables change, and a control point may subscribe to receive this information.

The service publishes updates by sending event messages. Event messages contain the names of one of more state variables and the current value of those variables. These messages are also expressed in XML and formatted using GENA.

A special initial event message is sent when a control point first subscribes; this event message contains the names and values for all evented variables and allows the subscriber to initialize its model of the state of the service.

To support multiple control points, all subscribers are sent all event messages, subscribers receive event messages for all evented variables, and event messages are sent no matter why the state variable changed (in response to an action request or due to a state change).

Presentation

If a device has a URL for presentation, then the control point can retrieve a page from this URL, load the page into a browser, and depending on the capabilities of the page, allow a user to control the device and/or view device status. The degree to which each of these can be accomplished depends on the specific capabilities of the presentation page and device.

In Summary:

- UPnP is based on wire protocols (just like the Internet), not APIs, allowing it to be truly media and platform agnostic.

- UPnP is based on existing standards, making interoperability easy to accomplish.

- UPnP has huge industry momentum, assuring success.

- While being standards based, UPnP is at the same time flexible and able to meet the needs of today's and the future's networked devices.

An Example UPnP Network

To further understand when and how each of the steps in UPnP networking takes place, it helps to define a small network with just a few UPnP devices. We can then describe how these devices interact as it relates to UPnP.

For this paper I have chosen a network that contains the following UPnP enabled devices:

An Internet Gateway. This device could be a standalone gateway device or a PC acting as a gateway. The device may be a control point, yet may not. Services provided by the device might include Internet access, a Dynamic Host Configuration Protocol (DHCP) server, a DNS proxy and a storage service. The gateway will also connect to several home LAN media and serve as a bridge for these media. Media used includes IEEE 802.11 Wireless, A Power Line network, A Phone Line network, and IEEE 1394.

Several Smart Appliances. For the purposes of this example, the network will contain several appliances that are UPnP enabled. This will include a clock radio, a coffee pot, and a microwave oven—all plugged into the power line network. The network will also contain a UPnP printer connected to the phone line network.

A Home Entertainment System. The home entertainment system includes several devices connected together via IEEE 1394 or 'Firewire' and connected to the gateway device. Components include a stereo system with a tuner, receiver, and CD jukebox player; a TV and VCR; and connections for additional A/V equipment such as a video or digital still camera. A new DVD jukebox player will also be added to this network.

A Wireless Enabled Laptop. The head of the household uses a laptop computer with a wireless network adapter at work, and occasionally brings this laptop home to finish up tasks left over from work.

Although there are many other possibilities for components participating in this network, the network is being kept relatively simple in this example to serve the purpose of explanation of the operation of UPnP.
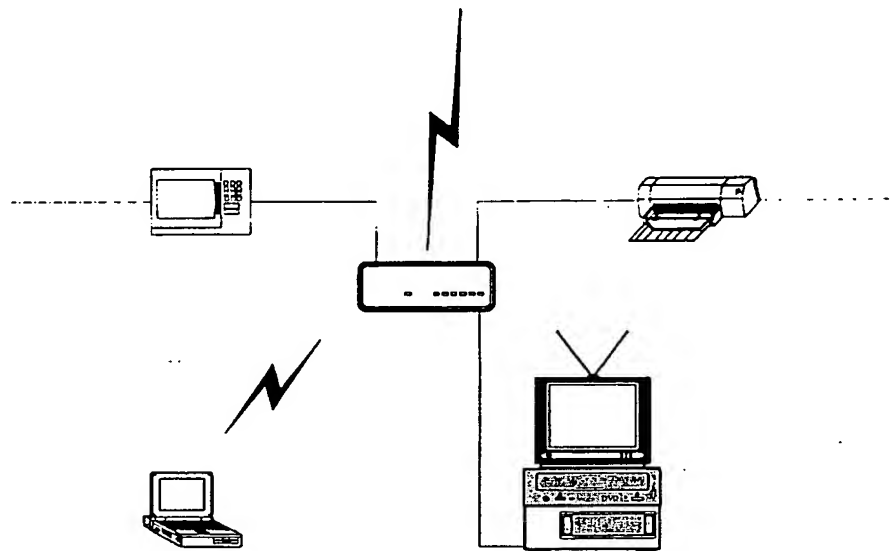
Figure 6: An Example Network

The Story Begins
To begin this scenario, all the components in the above network are up and running and have discovered each other using UPnP protocols, except for the laptop and DVD player.

Over dinner a couple of weeks ago the family discussed the success of DVD in the video market and how they were still not experiencing the high quality video they could, as they are stuck with only a video cassette player.

Mom received an invitation in the mail to join a DVD club. The family sat down and each picked several DVDs to start their collection. Today, mom called dad at work and told him the shipment of DVDs had arrived. Unfortunately they had neglected to purchase the key piece of equipment necessary to view them, the DVD player.

On his way home from work, dad (being the expert on such purchases) stopped by the electronic super store and purchased the latest in DVD jukebox players, making sure to get one that was UPnP enabled.

When dad arrived home, they unpacked the new toy and plugged it into the entertainment system using IEEE 1394. With UPnP, that was all that was needed for the rest of the household network to know of its existence.

Device Addressing

The first thing the new DVD player had to do was to obtain an address in order to participate on the network. Each device must have a DHCP client and search for a DHCP server when the device is first connected to the network.

If the DHCP client in the DVD player does not get a response from a server after waiting a short period of time, it will retry to ensure a server has a chance to respond. If the network does not contain a running DHCP server, the DVD player will use automatic IP addressing (Auto-IP) to choose an appropriate address.

With Auto-IP, the device intelligently chooses an IP address in the 169.254/16 range. The first and last 256 addresses in this range are reserved and must not be used. The selected address must then be tested to determine if the address is already in use. If the address is in use by another device, another address must be chosen and tested, up to an implementation dependent number of retries.

If the network has a DHCP server available, this whole process can take well under a second to complete. If the network does not have a DHCP server, requiring the device to use Auto-IP, the process will take a little longer. If the address is assigned using Auto-IP, the DVD player will periodically check to see if a DHCP server becomes available on the network to ensure connectivity is maintained among devices.

At this point, the DVD player either has an address assigned by the DHCP server, and all other devices in the network have an address in the same subnet, or the DVD player has an Auto-IP address. In either case, the DVD player can communicate with other devices on the network using TCP/IP.

Once the DVD player has a valid IP address for the network, it can be located and referenced on that network through that address. There may be situations where the end user needs to locate and identify a device. In these situations, a friendly name for the device is much easier for a human to use than an IP

address. However, the use of DNS for name to address mapping is outside the scope of UPnP.

Discovery - Advertisement

Now that our new device has an address and can communicate on the network, it needs to make itself known to those UPnP control points that were already up and running on the network. This is one form of discovery in UPnP. When a device is added to the network, the UPnP discovery protocol allows that device to advertise its services to control points on the network.

When a new device is added to the network, it multicasts discovery messages advertising its embedded devices and services. Any interested control point can listen to the standard multicast address for notifications that new services are available.

The discovery messages our DVD player sends will include a time stamp to indicate how long the advertisement is valid. Before this time expires, the DVD player must re-send its advertisement. Otherwise, control points can assume the device is no longer available. The DVD player should also send a message to explicitly tell the network it is going away before going offline.

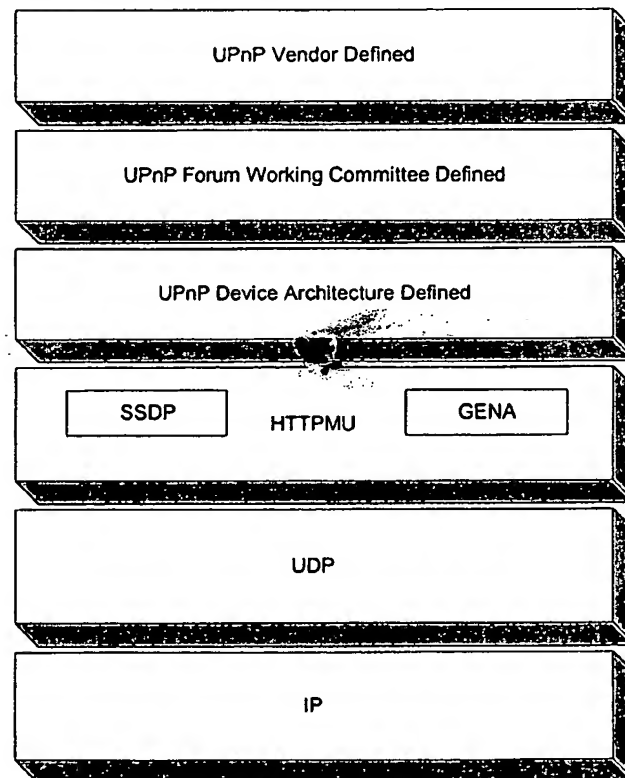The protocol stack shown here is used to send and receive advertisements.

Figure 7: Protocol Stack for Discovery Advertisements

Our DVD player, upon being plugged into the network, will send GENA advertisements for each device and service, announcing its presence. Since these messages are being delivered over UDP, an unreliable transport, they could be sent several times to ensure they are received by all interested control points.

Discovery - Search

With the hard work of connecting the DVD player done (dad also populated the jukebox with the DVDs they received in the mail), dad sits down on the couch with his laptop to finish up the presentation he is scheduled to deliver on Monday.

Dad's laptop is also UPnP enabled, so the addressing and discovery advertisements take place just as with the new DVD player. Dad begins his work as part of the home network with no additional configuration. Since it's Friday night, dinner is still an hour away, he doesn't have to have his presentation completed until Monday and there is a new toy in the house, dad wants to play.

Dad wants to start his favorite DVD movie playing and see how the new toy works. Of course he could try and figure out the new remote that came with the player, but since he's sitting on the couch with his laptop running already, and the DVD is available on the UPnP network, why get up?

Dad starts a video control application on his laptop. Starting this application brings a new control point into the network. All of the video devices on the network are displayed and dad selects the DVD player. Dad then selects the disc he wants to play and starts it rolling. He can also use the video control application to turn on the TV.

Several other steps in UPnP networking have just taken place. For the first time, a new control point has been brought into the network. When a new control point is added to the network, it multicasts a SSDP discovery message searching for interesting devices and services. All devices must listen to the standard multicast address for these messages and must respond if any of their embedded devices or services matches the search criteria in the discovery message. The video control application dad launched is looking specifically for video source devices.

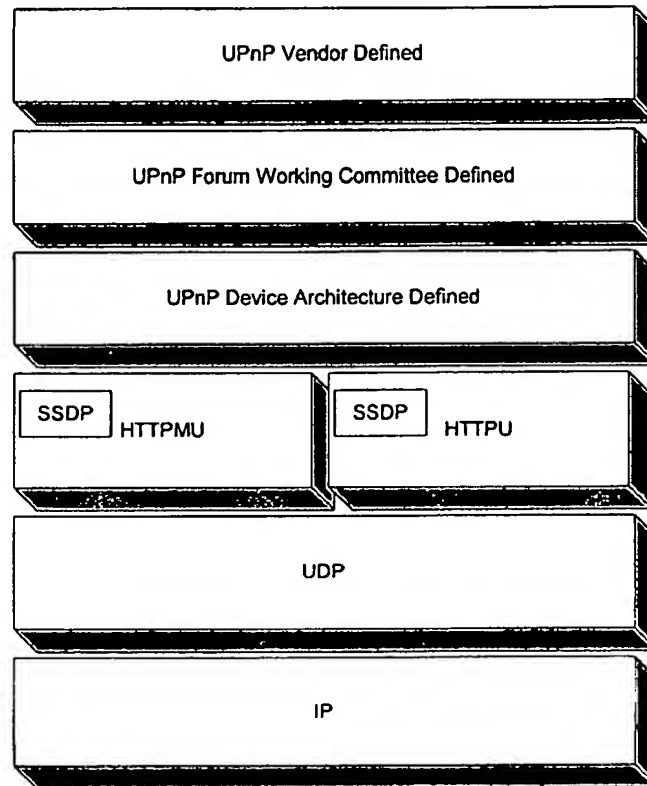The protocol stack used for these search messages is shown here:

Figure 8: Protocol Stack for Discovery Search

These search messages contain vendor-specific information, such as device or service types and identifiers. The device or service types defined by a UPnP working committee for these types of devices, in this case video source devices, are added. This information is encapsulated in a SSDP request sent using HTTPMU. Responses to these search requests will be sent using unicast UDP with SSDP headers.

The responses to these search requests contain the same information contained in discovery advertisements. These responses are sent to the IP address of the control point initiating the search, in this case dad's laptop.
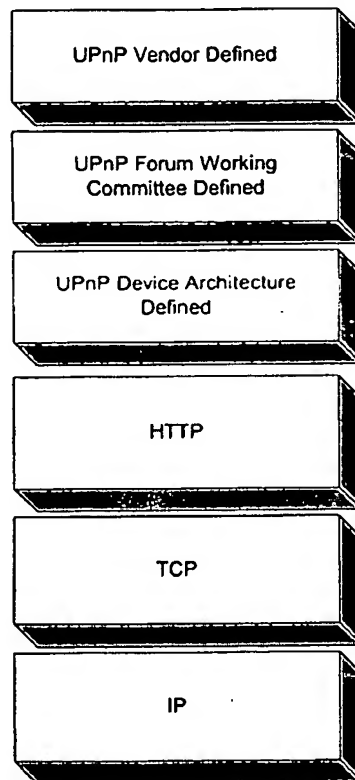
Description

The new control point running in dad's laptop now has information about all of the video source devices on the network. For the first time in this story we have a situation where a control point needs more information on a device, and thus we need to move into the description phase.

The responses received for the search discovery request contain the URL from which to obtain device descriptions.

To retrieve a UPnP device description, the control point issues an HTTP GET request on the URL from the discovery message, and the device returns the

device description. URLs for the service descriptions are part of the device description, consequently service descriptions can be retrieved in the same way. The protocol stack used for the description step is as follows:

Figure 9: Protocol Stack for Description

The UPnP description for a device is an XML document that contains several pieces of vendor-specific information, definitions of all embedded devices, URL for presentation of the device, and an enumeration of all services, including their URLs for control and events. UPnP vendors can extend the standard device and service description to include additional state variables, actions, and even whole services. In this way UPnP permits flexibility while adhering to basic

standards. Example device and service descriptions can be found in the UPnP device architecture document.

Presentation

The application running on dad's laptop can determine which devices and services to present and how to present them. Alternatively, if the DVD player hosts a presentation (web) page, this HTML page could be downloaded and used to control the device as well.

The URL for the presentation page is contained in the device description. Retrieving this page requires the control point to issue an HTTP GET request to the presentation URL. The device will then return a presentation page. The protocol stack used is as follows.
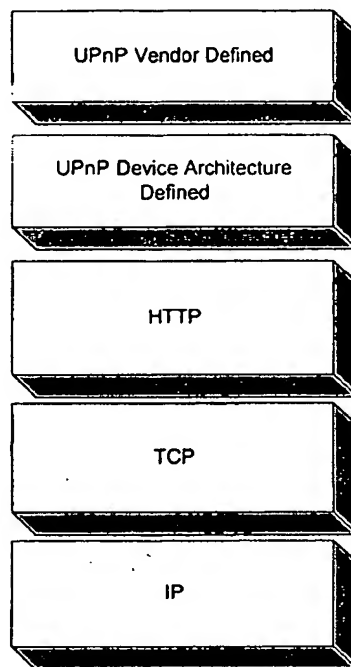

Figure 10: Protocol Stack for Presentation

The UPnP Device Architecture document specifies that this page be written in HTML. This is similar to web browsing; except here dad browses to the device to control it.

The capabilities of the presentation page are completely specified by the UPnP vendor. To implement a presentation page, a UPnP vendor may wish to use UPnP mechanisms for control and/or events, leveraging the device's existing capabilities. Notice that there is no UPnP Forum element defined in presentation, it is completely up to the vendor!

Control

Dad needs to control the DVD player, to select a DVD movie and start the player running. For this, he could use either the presentation page or the generic video control application.

Once the control point has knowledge of a device and its services, it can invoke actions on those services and get return values. At the same time, the control point can poll those services for the values of their state variables.

Invoking actions is a kind of remote procedure call; a control point sends the action to the device's service, and when the action has completed (or failed), the service returns any results or errors. The control point can also poll the value of state variables.

To control the DVD player, dad's laptop sends a control message to the control URL (contained within the device description) for the DVD's service. The DVD player service returns any results or errors from the action. The effects of the action may also be monitored by changes in the state variables of the service. These state variable changes are published to all interested control points as described in events, but the values of these state variables can be queried, which is a variant of a control request.

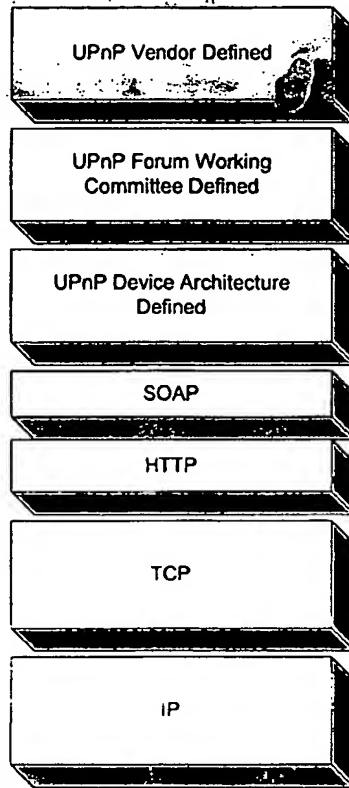The following protocol stack is used for control.

Figure 11: Protocol Stack for Control

The vendor specific information contained in control messages include argument values. The UPnP Forum working committee defines the action names, argument names and variables contained in these messages. This information is encapsulated in UPnP specific formats and formatted using SOAP, then transmitted using HTTP over TCP/IP.

The device must respond to the control request in 30 seconds. The response could indicate that the action is still pending and an event will signify completion.

Our control application may also want to query the state of a particular service variable, for example, our DVD player service may have a service with a state variable containing the run time of a particular DVD. Dad may want to know this so he knows how far into the movie he will be when dinner is served. Control points can also query state variables of a service, but only a single state variable with each query sent.

Revisiting Discovery, Description and Control

Now the DVD is playing and dad is ready to sit down and start work on his presentation. Looking at his new acquisition, he notices that the clock is flashing. Not only that, but the clock on the VCR is also flashing. He briefly considers getting the electrical tape and stopping the flashing the old fashioned way, but mom mentions that the clock on the microwave, coffee pot and alarm clock are also wrong, as they briefly lost power earlier in the afternoon.

Dad remembers that a clock set application came with his Internet gateway. What better time to try it out than now. He hadn't loaded the application on his laptop, but two factors have made it simple for this application to be found and run.

First of all, the gateway is providing storage services for the home network. It has a storage service making disk space available for the network. This time set application is available on the disk shared by the gateway. The second thing that enables this application to be run easily is that the operating system on Dad's laptop is UpnP-enabled, including his file browser. When he brings up the browser, it automatically searches the network for devices providing file storage services, and the storage in the gateway device appears.

Dad can now click on the clock set application and it can do the following:

- Locate the internet connection and connect to a time source on the Internet to get the official time.

- Use UPnP discovery to search the network for all devices providing clock services.

- Loop through each of the devices and submit 'set' actions to each of their clock services.

That was easy. Exploring further, dad finds that the clock application can be configured to run periodically from the gateway which is running as a control point itself. He sets the application up to run at 4:00 A.M. every morning and never has to worry about setting clocks again.

Eventing

It is getting close to dinnertime and dad has finished his presentation. He'd like to have a hardcopy so he can review it over dinner. Since a UPnP printer is connected to the phone line network in the kitchen, it is already available to his laptop through his printer browser.

Dad chooses to print to this printer and off it goes. He sets his laptop down and just starts to get engrossed in his movie when a popup appears on his laptop notifying him that the printer is out of ink. While this is possible today with printers directly connected to PCs, with UPnP the printer and print browser use UPnP eventing.

Dad was getting ready to call his son to ask him to change the ink cartridge, when his son walked by and told him that it was already taken care of. Just so happens that his son was doing homework on his PC in his room, when his PC received the same notification. All control points on a network that register for events receive the notifications.

The state variables described in a service description can be evented. The service publishes updates when these variables change. A control point, such as the print browser in this case, may subscribe to receive this information by sending a subscription message. The publisher of the event can accept this subscription and respond with a duration for the subscription. The subscriber can renew its subscription, or cancel the subscription when no longer interested.
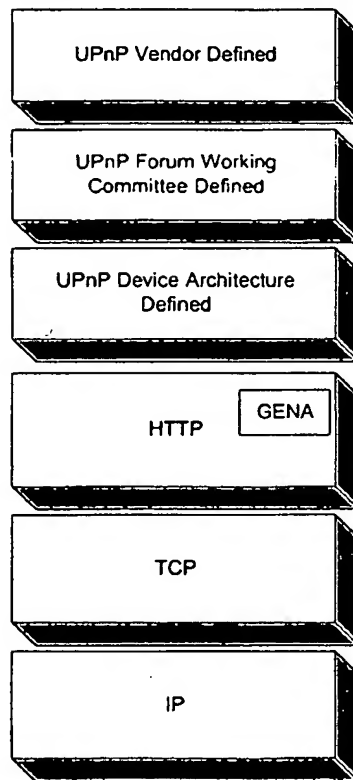
The following protocol stack is used for eventing:

Figure 12: Protocol Stack for Eventing

URLs for subscription, subscription durations, specific variable values, and variable names are formatted using GENA and delivered using TCP/IP.

Finishing Up

The son was working on his homework and getting ready to print, so he thought he would make his life easier by moving the printer to his room. He didn't think dad would notice, but a dialog popped up on dad's computer as soon as the son powered down the printer.

While this could have been done through eventing, in this case it was part of the discovery protocol. When a device leaves the network gracefully, it sends an advertisement that it is leaving. As a result, all control points will have perfect knowledge of network state.

Well, dinner is over, the movie is done and dad and mom are sitting on the couch. Mom notices dad's laptop is still running when he should be paying attention to her. Mom reaches to the laptop and pushes the button for mood control; the lights dim, the shades go down, soft music comes on, and the laptop shuts down

UPnP APIs

What APIs Should Be Implemented?

Because UPnP is based on standard protocols, and interoperability is achieved through using these common wire protocols, the APIs used to request UPnP services can be tailored to the platform the device is implemented on.

Windows devices such as PCs, Pocket PCs, and cell phones can use a common Windows API, whereas devices implemented on another platform can expose UPnP services through their own platform specific API. Regardless, these devices will interoperate on the wire protocol level, ensuring platform independence.

If applications are to be built on the services provided by the UPnP implementation, APIs should enable all facets of UPnP, namely:

- Discovery
- Description
- Control
- Eventing
- Presentation

Windows Me provides a rich API for all of the above and can be used to build or create UPnP applications for controlling multiple devices, as described in the scenarios earlier. For more information, please refer to the SDK.

Acronyms

API	Application Programming Interface
ARP	Address Resolution Protocol
DHCP	Dynamic Host Configuration Protocol
DNS	Domain Name System
FXPP	Flexible XML Processing Profile
GENA	General Event Notification Architecture
HTML	HyperText Markup Language
HTTPMU	HTTP Multicast over UDP
HTTPU	HTTP (unicast) over UDP
SOAP	Simple Object Access Protocol
SSDP	Simple Service Discovery Protocol
UPC	Universal Product Code
UPnP	Universal Plug and Play
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
URN	Uniform Resource Name
USN	Unique Service Name
UUID	Universally Unique Identifier
XML	Extensible Markup Language

Summary

Now, more than ever before, computing power is being added to smaller, more common devices. Inexpensive and ubiquitous networking media technologies are here, or are close to realization. The price drops in networking and computing power of recent years are great.

Combining computing power and connectivity in everyday devices with networking that is easy to use and configure is leading to new cumulative benefits: everyday tasks are easier to accomplish and people have more time to enjoy a higher quality of life. People are more connected to the world than ever before, a development that also risks people becoming overwhelmed. Thus tasks must be made easy in order for people to use them effectively.

Universal Plug and Play is an open initiative to take existing standards, existing technologies and existing knowledge, re-purpose it, and deliver on the promise and opportunity of the networked world. Standards-based, simple enough for the smallest appliances to implement, powerful enough to scale to the global Internet, and based on the proven approach of Internet protocols, Universal Plug and Play is an incremental approach, but an approach that has been proven to work.

For More Information

References

Universal Plug and Play Device Architecture

http://www.upnp.org/Device_Architecture_v0.92_.htm

RFC 2616

HTTP: Hypertext Transfer Protocol 1.1. IETF request for comments.

http://search.ietf.org/rfc/rfc2616.txt?number=2616

RFC 2279

UTF-8, a transformation format of ISO 10646 (character encoding). IETF request for comments.

http://search.ietf.org/rfc/rfc2279.txt?number=2279

XML

Extensible Markup Language. W3C recommendation.

http://www.w3.org/XML/

Auto-IP

Automatically Choosing an IP Address in an Ad-Hoc IPv4 Network. IETF draft.

http://search.ietf.org/internet-drafts/draft-ietf-dhc-ipv4-autoconfig-05.txt.

RFC1034

Domain Names - Concepts and Facilities. IETF request for comments.

http://search.ietf.org/rfc/rfc1034.txt?number=1034

RFC1035

Domain Names - Implementation and Specification. IETF request for comments.

http://search.ietf.org/rfc/rfc1035.txt?number=1035

RFC 2131

Dynamic Host Configuration Protocol. IETF request for comments.

http://search.ietf.org/rfc/rfc2131.txt?number=2131

RFC 2136

Dynamic Updates in the Domain Name System. IETF request for comments.

http://search.ietf.org/rfc/rfc2136.txt?number=2136

Dynamic DNS Updates by DHCP Clients and Servers

Interaction between DHCP and DNS. IETF Draft.

http://search.ietf.org/internet-drafts/draft-ietf-dhc-dhcp-dns-12.txt

GENA

General Event Notification Architecture. IETF Draft.

HTTPMU, HTTPU

HTTP Multicast over UDP, HTTP Unicast over UDP. IETF Draft.

SSDP

Simple Service Discovery Protocol. IETF Draft.

FXPP

Flexible XML Processing Profile.

Specifies that unknown XML elements and their sub elements must be ignored.
IETF draft.

RFC 1123

Includes format for dates, for, e.g., HTTP DATE header.

IETF request for comments.

http://search.ietf.org/rfc/rfc1123.txt?number=1123

RFC 1766

Format for language tag for, e.g., HTTP ACCEPT-LANGUAGE header.

IETF request for comments.

http://search.ietf.org/rfc/rfc1766.txt?number=1766

RFC 2387

Format for representing content type; for example, a mimetype element for an icon.

IETF request for comments.

http://www.ietf.org/rfc/rfc2387.txt?number=2387

UPC

Universal Product Code. 12-digit, all-numeric code that identifies the consumer package. Managed by the Uniform Code Council.

http://www.uc-council.org/main/ID_Numbers_and_Bar_Codes.html

XML

Extensible Markup Language. W3C recommendation.

http://www.w3.org/XML/

XML Schema (Part 1: Structures, Part 2: Datatypes)

Grammar defining UPnP Template Language. Defined using XML.

W3C working draft.

Part 1: Structures http://www.w3.org/TR/xmlschema-1/

Part 2: Datatypes http://www.w3.org/TR/xmlschema-2/

HTML

HyperText Markup Language. W3C recommendation.

http://www.w3.org/MarkUp/

HTTP Extension Framework

Describes a generic extension mechanism for HTTP.

W3C request for comments.

http://www.w3.org/Protocols/HTTP/ietf-http-ext/

SOAP

Simple Object Access Protocol.

Defines a protocol in XML, over HTTP, for remote procedure calls.

IETF draft and W3C Technical Report.